NVM Express Technical Errata

Errata ID	**018**
Change Date	**7/28/2011**
Affected Spec Ver.	**NVM Express 1.0b**
Corrected Spec Ver.	

Submission info

Name	Company	Date
Ken Okin	Virident	7/27/2011
Kevin Marks	Dell	7/27/2011

This erratum corrects a typo in the Completion Queue Head definition.

This erratum clarifies the status field returned in the Get Log Page entry.

Editorial changes are made to the first four commands in section 5.

Description of the specification technical flaw

Update the register definition in section 3.1.11 as shown below:

Bit	Type	Reset	Description
31:16	RO	0	Reserved
15:00	RW	0h	**Completion Queue Head (CQH):** Indicates the new value of the Completion Queue Head entry pointer. This value shall overwrite any previous Completion Queue Head value provided. The difference between the last CQH write and the current CQH entry pointer write indicates the number of entries that are now available for re-use by the controller in the Completion Queue. Note: ~~Competion~~ Completion Queue rollover needs to be accounted for.

Update Figure 58 as shown below:

Figure 58: Get Log Page – Error Information Log Entry

Bytes	Description
07:00	**Error Count:** This is a 64-bit incrementing error count, indicating a unique identifier for this error. The error count starts at 1h, is incremented for each unique error log entry, and is retained across power off conditions. A value of 0h indicates an invalid entry; this value may be used when there are lost entries or when there a fewer errors than the maximum number of entries the controller supports.
09:08	**Submission Queue ID:** This field indicates the Submission Queue Identifier of the command that the error information is associated with.
11:10	**Command ID:** This field indicates the Command Identifier of the command that the error is assocated with.
13:12	**Status ~~Code~~ Field:** This field indicates the Status ~~Code~~ Field ~~that~~ for the command that completed ~~with~~. The Status Field is located in bits 15:01; bit 00 corresponds to the Phase Tag posted for the command.
15:14	**Parameter Error Location:** This field indicates the byte and bit of the command parameter that the error is associated with, if applicable. If the parameter spans multiple bytes or bits, then the location indicates the first byte and bit of the parameter.

Bits	Description
7:0	Byte in command that contained the error. Valid values are 0 to 63.
10:8	Bit in command that contained the error. Valid values are 0 to7.
15:11	Reserved

Bytes	Description
23:16	**LBA:** This field indicates the first LBA that experienced the error condition, if applicable.
27:24	**Namespace:** This field indicates the namespace that the error is associated with, if applicable.
28	**Vendor Specific Information Available:** If there is additional vendor specific error information available, this field provides the log page identifier associated with that page. A value of 00h indicates that no additional information is available. Valid values are in the range of 80h to FFh.
63:29	Reserved

Update the section 5.1 as shown below:

The Abort command is used to ~~cancel/~~abort a specific command previously issued to the Admin Submission Queue or an I/O Submission Queue. Host software may have multiple Abort commands outstanding, subject to the constraints of the Abort Command Limit indicated in the Identify Controller data structure in Figure 65. An ~~abort~~ Abort command is a best effort command; the command to abort may have already completed, currently be in execution, or may be deeply queued. It is implementation specific if/when a controller chooses to complete the command when the command to abort is not found.

The Abort command uses the Command Dword 10 field. All other command specific fields are reserved.

Figure 26: Abort – Command Dword 10

Bit	Description
31:16	**Command Identifier (CID):** This field specifies the command identifier of the command to be aborted, that was specified in the CDW0.CID field within the command itself.
15:00	**Submission Queue Identifier (SQID):** This field specifies the identifier of the Submission Queue that the command to be aborted is associated with.

Update the first two paragraphs in section 5.1.1 as shown below:

A completion queue entry is posted to the Admin Completion Queue ~~when~~ if the command has been completed and a corresponding completion queue entry has been posted to the appropriate Admin or I/O Completion Queue. Dword 0 of the completion queue entry indicates whether the command was aborted. If the command was successfully aborted, then bit 0 of Dword 0 is cleared to ‘0’. If the command was not aborted, then bit 0 of Dword 0 is set to ‘1’.

Command specific errors associated with the Abort command are defined in Figure 27.

Update the first two paragraphs of section 5.2 as shown below:

Asynchronous events are used to notify host software of error and health information as these events occur. To enable asynchronous events to be reported by the controller, host software needs to issue one or more Asynchronous Event Request commands to the controller. The controller indicates an event to the host by completing an Asynchronous Event Request command. Host software should expect that the controller may not execute the command immediately; the command ~~is~~ should be completed when there is an event to be ~~posted~~ reported.

The Asynchronous Event Request command is issued by host software to enable the ~~signaling~~ reporting of asynchronous events from the controller. This command has no timeout. The controller posts a completion queue entry for this command when there is an asynchronous event to ~~signal~~ report to the host. If Asynchronous Event Request commands are outstanding when ~~a Submission Queue is deleted or~~ the controller is reset, the commands are aborted.

Update the last four paragraphs of section 5.2 as shown below:

Host software may issue multiple Asynchronous Event Request commands to reduce event ~~signaling~~ reporting latency. The total number of simultaneously outstanding Asynchronous Event Request commands is limited by the Asynchronous Event Request Limit specified in the Identify Controller data structure in Figure 65.

Asynchronous events are grouped into event types. The event type information is indicated in Dword 0 of the completion queue entry for the Asynchronous Event Request command. When the controller posts a completion queue entry for an outstanding Asynchronous Event Request command and thus ~~signals~~ reports an asynchronous event, subsequent events of that event type are automatically masked by the controller until the host clears that event. An event is cleared by reading the log page associated with that event using the Get Log Page command (see section 5.10).

The following ~~There are three~~ event types are defined:

- Error event: Indicates a general error that is not associated with a specific command. To clear this event, host software reads the Error Information log using the Get Log Page command.
- SMART / Health Status event: Indicates a SMART or health status event. To clear this event, host software reads the SMART / Health Information log using Get Log Page. The SMART / Health conditions that trigger asynchronous events may be configured in the Asynchronous Event Configuration feature using the Set Features command (see section 5.12).
- Vendor Specific event: Indicates a vendor specific event. To clear this event, host software reads the indicated vendor specific log page using Get Log Page.

When the controller needs to ~~signal~~ report an event and there are no outstanding Asynchronous Event Request commands, the controller queues the event internal to the controller and ~~signals~~ reports it when an Asynchronous Event Request command is ~~issued~~ received.

Update the first paragraph of section 5.2 .1 as shown below:

A completion queue entry is posted to the Admin Completion Queue ~~when~~ if there is an asynchronous event to ~~signal~~ report to the host. For the Asynchronous Event Request command, there are some Command Specific Errors that are specific to this command that are defined in Figure 28.

Update Figure 29 as shown below:

Figure 1: Asynchronous Event Request – Completion Queue Entry Dword 0

Bit	Description
31:24	Reserved
23:16	**Associated Log Page:** Indicates the log page associated with the asynchronous event. This log page needs to be read by the host to clear the event.
15:08	**Asynchronous Event Information:** Refer to Figure and **Error! Reference source not found.** for detailed information regarding the asynchronous event.
07:03	Reserved
02:00	**Asynchronous Event Type:** Indicates the type of the asynchronous event. ~~The Error status type indicates an error condition for the controller. The SMART / Health status type provides a controller or NVM health indication.~~ More specific information on the event is provided in the Asynchronous Event Information field. Value | Definition 0h | Error status 1h | SMART / Health status 2h – 6h | Reserved 7h | Vendor specific

Update Figure 30 as shown below:

Figure 30: Asynchronous Event Information – Error Status

Value	Description
0h	**Invalid Submission Queue:** Host software ~~Software~~ wrote the doorbell of a queue that was not created.
1h	**Invalid Doorbell Write Value:** Host software ~~Software~~ attempted to write an invalid doorbell value. Some possible causes of this error are: • the value written was out of range of the corresponding queue's base address and size, • the value written is the same as the previously written doorbell value, • host software attempts to add a command to a full Submission Queue, and • host software attempts to remove a completion queue entry from an empty Completion Queue.
2h	**Diagnostic Failure:** A diagnostic failure was detected. This may include a self-test operation.
3h	**Persistent Internal Device Error:** A failure occurred within the device that is persistent or the device is unable to isolate to a specific set of commands. If this error is indicated, then the CSTS.CFS bit may be set to '1' and the host should perform a reset as described in section 7.3.
4h	**Transient Internal Device Error:** A transient error occurred within the device that is specific to a particular set of commands and operation may continue.
5h	**Firmware Image Load Error:** The firmware image could not be loaded. The controller reverted to the previously active firmware image or a baseline read-only firmware image.
6h - FFh	Reserved

Update the first paragraph of section 5.3 as shown below:

The Create I/O Completion Queue command is used to create all I/O Completion Queues with the exception of the Admin Completion Queue. The Admin Completion Queue is created by specifying its base address in the ACQ register. If a PRP List is provided to describe the CQ, then the PRP List shall be maintained by host software at the same location in host physical memory and the values in the PRP List shall not be modified until the corresponding Delete I/O Completion Queue command for this CQ is completed successfully or the controller is reset. If the PRP List values are modified, the behavior is undefined.

Update Figure 33 as shown below:

Figure 33: Create I/O Completion Queue – Command Dword 10

Bit	Description
31:16	**Queue Size (QSIZE):** This field indicates the size of the Completion Queue to be created. Refer to section 4.1.3. This is a 0's based value.
15:00	**Queue Identifier (QID):** This field indicates the identifier to assign to the Completion Queue to be created. This identifier corresponds to the Completion Queue Head Doorbell used for this command (i.e., the value *y* in section 3.1.11). This value shall not exceed the value reported in the Number of Queues feature (see section 5.12.1.7) for I/O Completion Queues.

Modify section 5.3.1 as shown below:

If ~~When~~ the command is completed, then the controller shall post a completion queue entry to the Admin Completion Queue indicating the status for the command.

Create I/O Completion Queue command specific errors are defined in Figure 35.

Figure 35: Create I/O Completion Queue – Command Specific Errors Values

Value	Description
1h	**Invalid Queue Identifier:** The creation of the I/O Completion Queue failed due to an invalid queue identifier specified as part of the command.
2h	**Maximum Queue Size Exceeded:** ~~Software~~ The host attempted to create an I/O Completion Queue with a number of entries that exceeds the maximum supported by the controller, specified in CAP.MQES.
8h	**Invalid Interrupt Vector:** The creation of the I/O Completion Queue failed due to an invalid interrupt vector specified as part of the command.

Modify the first paragraph of section 5.4 as shown below:

The Create I/O Submission Queue command is used to create ~~all~~ I/O Submission Queues ~~with the exception of the Admin Submission Queue~~. The Admin Submission Queue is created by specifying its base address in the ASQ register. If a PRP List is provided to describe the SQ, then the PRP List shall be maintained by host software at the same location in host physical memory and the values in the PRP List shall not be modified until the corresponding Delete I/O Submission Queue command for this SQ is completed or the controller is reset. If the PRP List values are modified, the behavior is undefined.

Update Figure 37 as shown below:

Figure 37: Create I/O Submission Queue – Command Dword 10

Bit	Description
31:16	**Queue Size (QSIZE):** This field indicates the size of the Submission Queue to be created. Refer to section 4.1.3. This is a 0's based value.
15:00	**Queue Identifier (QID):** This field indicates the identifier to assign to the Submission Queue to be created. This identifier corresponds to the Submission Queue Tail Doorbell used for this command (i.e., the value *y* in section 3.1.10). This value shall not exceed the value reported in the Number of Queues feature (see section 5.12.1.7) for I/O Submission Queues.

Modify Figure 38 as shown below:

Figure 38: Create I/O Submission Queue – Command Dword 11

Bit	Description
31:16	**Completion Queue Identifier (CQID):** This field indicates the identifier of the Completion Queue to utilize for any command completions entries associated with this Submission Queue. The value of 0h (Admin Completion Queue) shall not be specified.
15:03	Reserved
02:01	**Queue Priority (QPRIO):** This field indicates the priority service class to use for commands within this Submission Queue. This field is only used when the weighted round robin with an urgent priority service class is the arbitration mechanism is selected; the field is ignored if weighted round robin with an urgent priority service class is not used. Refer to section 4.7. Value / Definition: 00b – Urgent 01b – High 10b – Medium 11b – Low
00	**Physically Contiguous (PC):** If set to '1', then the Submission Queue is physically contiguous and PRP Entry 1 (PRP1) is the address of a contiguous physical buffer. If cleared to '0', then the Submission Queue is not physically contiguous and PRP Entry 1 (PRP1) is a PRP List pointer.

Value	Definition
00b	Urgent
01b	High
10b	Medium
11b	Low

Modify section 5.4.1 as shown below:

When the command is completed, the controller ~~shall~~ posts a completion queue entry to the Admin Completion Queue indicating the status for the command.

Create I/O Submission Queue command specific errors are defined in Figure 39.

Figure 39: Create I/O Submission Queue – Command Specific Errors Values

Value	Description
0h	**Completion Queue Invalid:** The Completion Queue identifier specified in the command does not exist.
1h	**Invalid Queue Identifier:** The creation of the I/O Submission Queue failed due an invalid queue identifier specified as part of the command.
2h	**Maximum Queue Size Exceeded:** Host software ~~Software~~ attempted to create an I/O Submission Queue with a number of entries that exceeds the maximum supported by the controller, specified in CAP.MQES.

Disposition log

7/27/2011	Erratum captured.
7/28/2011	Updated vendor specific log entry clearing.
9/26/2011	Erratum ratified.